Temporal Gradient-Domain Path Tracing

Marco Manzi[1,⋆] Markus Kettunen[2,⋆] Frédo Durand[4] Matthias Zwicker[1] Jaakko Lehtinen[2,3]
[1]University of Bern [2]Aalto University [3]NVIDIA [4] MIT CSAIL
⋆ *Both authors contributed equally to this work.*

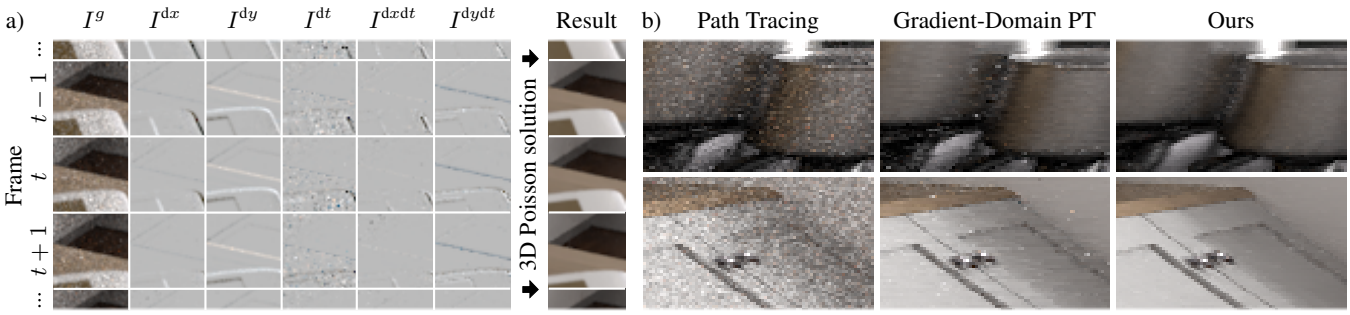


Figure 1: a) *In addition to standard path sampling, our method also estimates spatial, temporal and mixed finite differences for the frames of an animation. We then solve a 3D screened Poisson problem to reconstruct the animation whose frames best match the sampled data.* b) *Equal-time rolling shutter crops of the animation* KITCHEN 2. *The rows are extracted from sequential animation frames. Our method often reduces both spatial variance, seen as horizontal noise, and flickering, seen as vertical noise.*

Abstract

We present a novel approach to improve temporal coherence in Monte Carlo renderings of animation sequences. Unlike other approaches that exploit temporal coherence in a post-process, our technique does so already during sampling. Building on previous gradient-domain rendering techniques that sample finite differences over the image plane, we introduce temporal finite differences and formulate a corresponding 3D spatio-temporal screened Poisson reconstruction problem that is solved over windowed batches of several frames simultaneously. We further extend our approach to include second order, mixed spatio-temporal differences, an improved technique to compute temporal differences exploiting motion vectors, and adaptive sampling. Our algorithm can be built on a gradient-domain path tracer without large modifications. In particular, we do not require the ability to evaluate animation paths over multiple frames. We demonstrate that our approach effectively reduces temporal flickering in animation sequences, significantly improving the visual quality compared to both path tracing and gradient-domain rendering of individual frames.

Keywords: Monte Carlo rendering, gradient-domain rendering

Concepts: •**Computing methodologies** → *Ray tracing;*

1 Introduction

Monte Carlo path tracing is establishing itself as the algorithm of choice for movie production because of its physical realism and predictable behavior [Keller et al. 2015]. Rendering animations with hundreds of thousands of frames is still challenging, however, due to the significant computational effort it takes to reduce variance (noise) to acceptable levels. Current production pipelines employ conventional Monte Carlo rendering techniques that produce each frame separately, and apply post-process noise reduction filters. This seems wasteful, as the similarity between temporally nearby images is not exploited during the rendering process.

In this paper, we introduce a Monte Carlo technique that exploits temporal coherence during both rendering and reconstruction, as opposed to just a post-process, and yields consistent results without heuristic blending of samples across frames or other model-based reasoning. These properties contrast previous algorithms, including sample reprojection, spatio-temporal filtering, and the construction of smooth spatio-temporal function bases for the frames.

Our key idea is to sample the *differences* between corresponding pixels in temporally adjacent frames using correlated pairs of paths, and then integrate the Monte Carlo difference estimates across time. This effectively suppresses flickering, thanks to the surprising property, shown previously in gradient-domain path tracing [Kettunen et al. 2015], that correlated difference estimates followed by integration yields significant variance reduction for smooth signals.

Technically, we build on gradient-domain path tracing (GPT) [Kettunen et al. 2015], which samples spatial finite difference image gradients and reconstructs output images by solving a screened Poisson problem. We extend both the sampling and reconstruction steps into the three-dimensional spatio-temporal domain. We estimate the temporal differences using pairs of paths that share the same primary sample space coordinates in adjacent frames, further leveraging motion vectors provided by most standard path tracers to try to ensure that both paths share a similar primary hit point of the camera ray. This reduces variance in temporal differences, and hence increases quality, while remaining consistent. Finally, we introduce an adaptive sampling technique that exploits the sparsity of high variance regions in the gradient domain. Our method is lightweight and largely non-intrusive in the sense that it can be implemented relatively easily on top of an existing path tracer.

SA '16 Technical Papers, December 05-08, 2016, Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2980256

ACM Reference Format
Manzi, M., Kettunen, M., Durand, F., Zwicker, M., Lehtinen, J. 2016. Temporal Gradient-Domain Path Tracing. ACM Trans. Graph. 35, 6, Article 246 (November 2016), 9 pages. DOI = 10.1145/2980179.2980256 http://doi.acm.org/10.1145/2980179.2980256.

2 Related Work

Exploiting Temporal Coherence. While there is a broad literature on taking advantage of spatial coherence through filtering [Rushmeier and Ward 1994] or interpolation [Ward et al. 1988] to accelerate Monte Carlo rendering of individual images, there has been much less work on temporal coherence. Early approaches, as the one by Chen [1990] or Nimmeroff et al. [1996], relied on radiosity [Goral et al. 1984], and they focused on incrementally updating or interpolating radiosity solutions over time. Temporal coherence methods are widely studied in real-time rendering [Scherzer et al. 2012], but interactivity is outside the scope of this paper.

In the context of Monte Carlo techniques, Havran et al. [2003] first proposed to exploit temporal coherence in animations by updating and reprojecting samples to different points in time. Similarly, in final gathering in photon mapping [Tawara et al. 2004] and in irradiance caching [Smyk et al. 2005], final gather or irradiance samples can be sparsely updated and interpolated over time. An alternative strategy is to remove noise from image sequences produced by Monte Carlo rendering after the fact, which can be achieved using various image space filtering techniques. McCool [1999] proposed to use anisotropic diffusion, for example. He may have been the first to mention an extension to 3D spatio-temporal denoising for Monte Carlo rendering, although this was not demonstrated in his work. Meyer and Anderson [2006] use PCA analysis to construct a smooth basis for a sequence of images, then they project noisy images onto this basis for denoising. Recently, various image space filters have also been demonstrated in the spatio-temporal setting [Sen and Darabi 2012; Li et al. 2012; Rousselle et al. 2013; Moon et al. 2014]. Zimmer et al. [2015] further propose a path space decomposition approach and sophisticated motion estimation techniques to maximize temporal coherence.

A key difference to our work is that we take advantage of temporal coherence during both Monte Carlo sampling as well as reconstruction, instead of just filtering in a post-processing step. Furthermore, we sample temporal differences in an unbiased fashion, and our reconstruction can produce unbiased output if desired. This is not possible with any of the previous methods that use temporal coherence. In practice, though, an outlier-suppressing L_1 reconstruction is preferred over the unbiased method, like with previous gradient-domain rendering algorithms.

Gradient-domain Rendering. Gradient-domain rendering relies on sampling finite difference image gradients in addition to pixel intensities, where a gradient sample is the difference between a *base* and an *offset* light path through neighboring pixels. Offset paths are generated in a correlated fashion by shifting a base path sampled by a standard path tracer to a neighboring pixel, such that the two paths remain as similar as possible. Consequently, the magnitude of the difference in their throughputs is generally smaller than their individual contributions. Due to this reduced variance, output images after screened Poisson reconstruction are of higher quality compared to conventional rendering. Finally, gradient-domain rendering is unbiased if reconstruction employs the L_2 norm.

Lehtinen et al. [2013] originally introduced gradient-domain rendering for Metropolis light transport [Veach and Guibas 1997], and Manzi et al. [2014] proposed more general gradient sampling techniques to improve the original approach. Kettunen et al. [2015] showed that gradient sampling is also beneficial in conventional path tracing, and back their empirical results up with an analysis that studies the gradient sampling and reconstruction process through Fourier theory. Manzi et al. [2015] extend gradient-domain path tracing to bidirectional path tracing, combining the advantages of gradient and bidirectional sampling.

At its core, gradient-domain sampling estimates differences $\Delta_{i,j}$ between the intensities of neighboring pixels i and j as

$$\begin{aligned}\Delta_{i,j} &= \left(h(x) * \int_\Omega f(x,\bar{p}) - f(T_{ij}(x,\bar{p}))\left|T'_{ij}\right| \mathrm{d}\mu(\bar{p})\right)(x_i) \\ &= \left(h(x) * \int_\Omega g_{ij}(x,\bar{p})\mathrm{d}\mu(\bar{p})\right)(x_i), \end{aligned} \quad (1)$$

where x is the image coordinate, $h(x)$ a pixel filter, Ω the space of light paths, $(x,\bar{p})$ a path with additional parameters $\bar{p}$, and f the image contribution function. We call T_{ij} a *shift mapping* that maps a base to an offset path, and $|T'| = |\partial T/\partial\bar{x}|$ is the determinant of the Jacobian of $T(\bar{x})$. Gradient-domain rendering techniques may use various methods to sample base paths $(x,\bar{p})$ in Equation 1, such as unidirectional [Kettunen et al. 2015] or bidirectional path tracing [Manzi et al. 2015]. While estimating the horizontal and vertical finite differences, they also sample the image itself in a conventional manner. Finally, finite differences are sampled in both directions, that is, for each $\Delta_{i,j}$ also $\Delta_{j,i}$ is sampled, and the two are combined using multiple importance sampling (MIS).

Depending whether i and j are horizontal or vertical neighbors, let us classify the differences $\Delta_{i,j}$ into horizontal and vertical, and unroll them into two vectors I^{dx} and I^{dy}. In addition, let I^g denote the conventional *primary image* unrolled into a vector. Screened Poisson reconstruction solves for an image I that satisfies

$$I = \underset{\bar{I}}{\operatorname{argmin}} \left\|\alpha(\bar{I} - I^g)\right\|^p + \left\| \begin{pmatrix} H^{\mathrm{d}x}\bar{I} \\ H^{\mathrm{d}y}\bar{I} \end{pmatrix} - \begin{pmatrix} I^{\mathrm{d}x} \\ I^{\mathrm{d}y} \end{pmatrix} \right\|^p, \quad (2)$$

where α weights the influence of the pixel and gradient constraints, and $H^{\mathrm{d}x}$ and $H^{\mathrm{d}y}$ denote the horizontal and vertical finite difference operators on the 2D pixel grid. The solution image I simultaneously minimizes the pixelwise difference to the primary image, as well as the difference of its gradient to the sampled gradients. The solution of Equation 2 under the L_2 norm ($p = 2$) is unbiased, but often suffers from visual artifacts. The L_1 norm ($p = 1$) yields more pleasing results, at the cost of introducing bias.

3 Temporal GPT

Our temporal gradient-domain path tracing algorithm introduces three main conceptual components: finite differences over both space and time, a temporal shift mapping to obtain these gradients, and spatio-temporal screened Poisson reconstruction.

3.1 Space-Time Finite Differences

We sample three kinds of finite differences over space and time as illustrated in Figure 2:

1. regular image gradients over pairs of pixels in individual frames;
2. the temporal differences for a pixel between neighboring frames; and
3. second-order mixed space-time differences, that is, temporal differences between spatial gradients.

Spatial image gradients are computed precisely as in gradient-domain path tracing [Kettunen et al. 2015], and we will not review its *spatial shift mapping* here. In particular, we also evaluate spatial shifts across pixels in both forward and backward directions, and weight them using MIS. For clarity, our figures only feature the positive spatial shift. We call the shift mapping between frames the *temporal shift*, and the temporal differences consist of base-offset

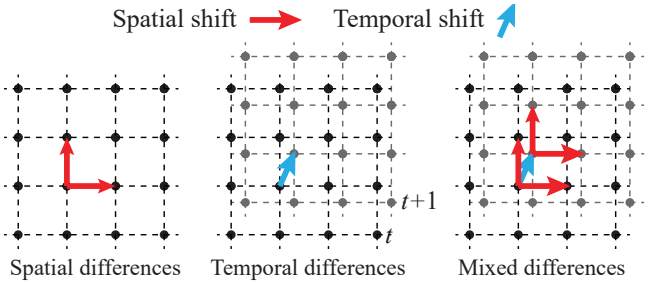


Figure 2: *Spatial, temporal, and mixed spatio-temporal finite differences over adjacent pixels in both space and time. Temporal finite differences are taken between paths in adjacent frames at times t and $t+1$. The mixed differences are second order, that is, temporal differences between spatial gradients in adjacent frames.*

path pairs in temporally adjacent frames. We compute the mixed differences after-the-fact by subtracting corresponding spatial gradients between adjacent frames.

3.2 Primary Sample Space Temporal Shift Mapping

For temporal and mixed differences, we propose a temporal shift mapping that copies the primary sample space [Kelemen et al. 2002] coordinates of the base path to the next frame, just with the time coordinate incremented by one full frame. This simply means the temporal offset path uses, with the exception of time, the same values for the uniform random variables that specify the base path.

The main advantage of this shift is its simplicity. The shift has unit Jacobian by construction, and as long as the renderer is deterministic, implementation is trivial. We can render each frame twice, as a *base frame* and a *temporal offset frame*. A base frame is rendered with conventional GPT as usual. The temporal offset frame is also rendered with GPT, by re-using the same random variables for each pixel as the previous base frame. Hence, subtracting a base frame from its succeeding temporal offset frame yields the temporal and mixed differences.

We illustrate this process in Figure 3. The red samples form the base frames, and the blue samples the temporal offset frames. Both base and temporal offset frames include a primary image (consisting of the samples indicated with filled circles) and spatial gradient images (consisting of the differences between the spatial offset samples and their corresponding primary samples).

The above scheme reduces the overhead of our approach by re-using computation similar to the original GPT algorithm. Specifically, for each base path (filled red circles) that contributes to the primal image in the base frame, we re-use it as the base path for our spatial gradients as in GPT. In addition, we re-use it as the base path for a temporal difference. Finally, we re-use the temporal offset paths, which we computed for the temporal differences, to again compute spatial gradients which we also use for the mixed gradients. Kettunen et al. [2015] report an overhead of $2.5\times$ of GPT over conventional path tracing. Our approach amounts to running GPT with half the samples for both the base and offset image in each frame, and hence we inherit the overhead of $2.5\times$.

A potential disadvantage of the primary sample space shift is that the path throughput function is often less smooth over primary sample space coordinates than other parameterizations, potentially leading to higher variance than the more sophisticated spatial shift that reuses path vertex positions instead. We address this by leveraging motion vectors (Section 4.2) to regain smoothness.

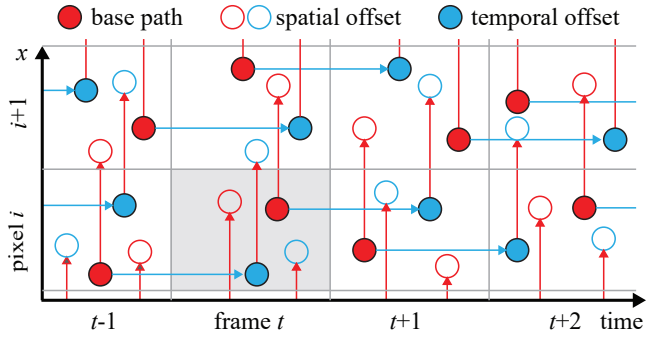


Figure 3: *We illustrate sampling of spatial, temporal, and mixed-spatio-temporal differences, using spatial (red) and temporal shifts (blue arrows). Our approach amounts to rendering each frame twice using GPT, as a base frame (red samples) and as a temporal offset frame (blue samples). Each offset frame has identical primary sample space values as its preceding base frame, which implements our temporal shift.*

3.3 Spatio-temporal Reconstruction

Spatio-temporal screened Poisson reconstruction is a direct extension of the 2D case (Equation 2). In addition to spatial gradients, we now also take into account the temporal and mixed spatio-temporal differences, and instead of reconstructing a single frame $\bar{I}$, reconstruct a sequence of frames $\bar{\mathbf{I}}$ simultaneously. Denoting the sequence of sampled conventional images by $\mathbf{I}^g$, and the sequences of sampled spatial, temporal, and spatio-temporal differences by $\mathbf{I}^{\mathrm{d}x}, \mathbf{I}^{\mathrm{d}y}, \mathbf{I}^{\mathrm{d}t}, \mathbf{I}^{\mathrm{d}x\mathrm{d}t}$, and $\mathbf{I}^{\mathrm{d}y\mathrm{d}t}$, we formulate spatio-temporal reconstruction as

$$\mathbf{I} = \underset{\bar{\mathbf{I}}}{\operatorname{argmin}} \left\|\alpha(\bar{\mathbf{I}} - \mathbf{I}^g)\right\|^p + \left\| \begin{pmatrix} H^{\mathrm{d}x}\bar{\mathbf{I}} \\ H^{\mathrm{d}y}\bar{\mathbf{I}} \\ H^{\mathrm{d}t}\bar{\mathbf{I}} \\ H^{\mathrm{d}x\mathrm{d}t}\bar{\mathbf{I}} \\ H^{\mathrm{d}y\mathrm{d}t}\bar{\mathbf{I}} \end{pmatrix} - \begin{pmatrix} \mathbf{I}^{\mathrm{d}x} \\ \mathbf{I}^{\mathrm{d}y} \\ \mathbf{I}^{\mathrm{d}t} \\ \mathbf{I}^{\mathrm{d}x\mathrm{d}t} \\ \mathbf{I}^{\mathrm{d}y\mathrm{d}t} \end{pmatrix} \right\|^p, \quad (3)$$

where $H^{\mathrm{d}x}$ and $H^{\mathrm{d}y}$ are the spatial finite difference operators on the pixel grid, $H^{\mathrm{d}t}$ is the temporal finite difference operator between adjacent frames, and $H^{\mathrm{d}x\mathrm{d}t}$ and $H^{\mathrm{d}y\mathrm{d}t}$ are the mixed spatio-temporal finite difference operators. These are simply the concatenations of the temporal and spatial operators, $H^{\mathrm{d}x\mathrm{d}t} = H^{\mathrm{d}t}H^{\mathrm{d}x}$ and $H^{\mathrm{d}y\mathrm{d}t} = H^{\mathrm{d}t}H^{\mathrm{d}y}$.

To process arbitrarily long input sequences we split them into overlapping subsequences, reconstruct each subsequence separately, and smoothly blend between overlapping temporal regions after reconstruction. Our approach is similar to the temporal extensions of screened Poisson reconstruction used by Bonneel et al. [2014; 2015]. However, they perform reconstruction in a causal manner, processing subsequent frames one-by-one over time.

We solve Equation 3 via its normal equations $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$ and a conjugate gradient solver in CUDA. The vector $\mathbf{x}$ consists of all pixels in all frames, and matrix $\mathbf{A}$ is sparse and its rows represent all the constraints in Equation 3. For efficiency our solver computes $\mathbf{A}^T\mathbf{A}$ on the fly without storing $\mathbf{A}$ explicitly.

3.4 Frequency Analysis

In analogy to the purely spatial case studied in previous work, the expected main benefit of adding temporal finite differences to the sampling and reconstruction process is the reduction of high frequency temporal variance (distracting flickering). To understand

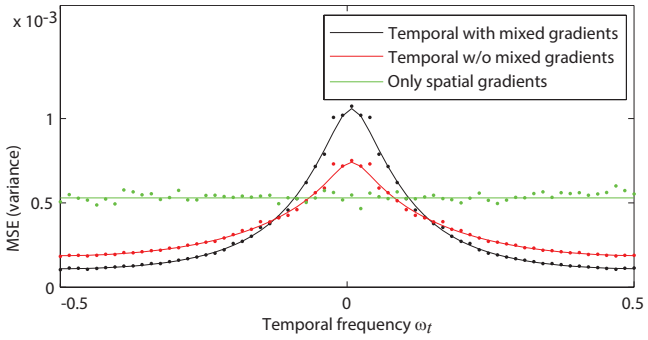


Figure 4: *The reconstruction error over temporal frequencies ω_t for different methods. The plot is based on an assumed standard error of the sampled conventional image $|\epsilon_F| = 0.1$, standard error of sampled gradients $|\epsilon_G| = 0.02$, and $\alpha = 0.2$, which is a typical scenario in practice. The dotted lines are from an empirical experiment to confirm the theory, where we added Gaussian noise with known variance to a sequence of images and their spatial and temporal differences, and then performed reconstruction numerically. The plots show an equal time comparison based on implementation details as described in Section 5.*

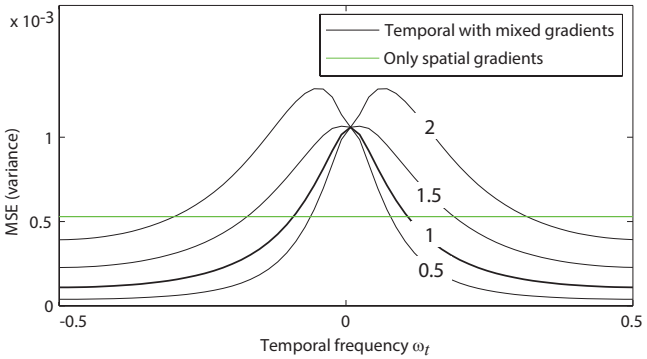


Figure 5: *We show the effect of different ratios of variances in temporal and mixed versus spatial gradients on the error in temporal frequencies. We plot ratios $\{0.5, 1, 1.5, 2\}$. Even if temporal and mixed gradients have more variance than spatial ones, which can be caused by fast camera or object motion, we still obtain high frequency noise reduction.*

this better, we extend the frequency analysis of gradient-domain path tracing presented by Kettunen et al. [2015].

For simplicity, we assume that the mean squared errors (MSEs) of the sampled spatial, temporal, and spatio-temporal gradients are all equal, and denote them by $|\epsilon_G|^2$. Note that these directly correspond to their variances. We denote the MSE of the sampled conventional image by $|\epsilon_F|^2$. Since Monte Carlo sampling produces white noise, the variances are identical across frequencies, simplifying the analysis. Kettunen et al. derive the relation between $|\epsilon_G|^2$ and $|\epsilon_F|^2$ under suitable simplifying assumptions, and show that typically sampled gradients have lower variance than sampled pixels. We take this as given.

Let us denote the Frequency domain MSE of the final screened L_2 Poisson reconstruction of the spatio-temporal animation sequence by $|\epsilon_{R_\alpha}(\omega_x, \omega_y, \omega_t)|^2$. Following the analysis of Kettunen et al., it is easy to see that in contrast to the errors $|\epsilon_F|^2$ and $|\epsilon_G|^2$ in the sampled base image and its gradients, the final reconstruction error *does* vary with spatio-temporal frequency ω_x, ω_y and ω_t. Introducing our temporal and mixed differences into their final reconstruction error, we get

$$|\epsilon_{R_\alpha}(\omega_x, \omega_y, \omega_t)|^2 = \frac{\alpha^4|\epsilon_F|^2 + |\epsilon_G|^2(|D^x|^2 + |D^y|^2 + |D^t|^2 + |D^{tx}|^2 + |D^{ty}|^2)}{(\alpha^2 + |D^x|^2 + |D^y|^2 + |D^t|^2 + |D^{tx}|^2 + |D^{ty}|^2)^2}, \quad (4)$$

where $|D^x(\omega_x, \omega_y, \omega_t)|^2 = 2 - 2\cos(2\pi\omega_x)$ is the power spectrum of the finite difference operator along horizontal spatial frequencies ω_x, and we omitted the arguments in the equation above for brevity. The spectra of the other finite difference operators along vertical spatial $|D^y|^2$ and temporal dimensions $|D^t|$ are defined analogously. The power spectrum of the mixed finite difference operator $|D^{tx}|^2$ is the product $|D^{tx}|^2 = (2 - 2\cos(2\pi\omega_x))(2 - 2\cos(2\pi\omega_t))$, and similar for $|D^{ty}|^2$. The error of screened Poisson reconstruction without temporal and mixed gradients, as in Kettunen et al. [2015], corresponds to Equation 4 without the $|D^t|^2, |D^{tx}|^2$, and $|D^{ty}|^2$ terms.

To gain insight, Figure 4 plots the reconstruction error from Equation 4 over temporal frequencies $|\epsilon_{R_\alpha}(\omega_t)|^2$, with spatial dimensions averaged over, assuming equal time taken for sampling. We compare standard GPT without temporal gradients (green) [Kettunen et al. 2015], temporal GPT with only first-order temporal differences $\mathbf{I}^{\mathrm{d}t}$ (red), and the full temporal GPT including also mixed differences $\mathbf{I}^{\mathrm{d}x\mathrm{d}t}, \mathbf{I}^{\mathrm{d}y\mathrm{d}t}$ (black). Note that all algorithms include standard spatial gradients. The comparisons are produced by omitting the respective terms in Equation 4.

We see that without temporal differences, the error is white noise over time (green curve). This is expected, as the frames are sampled independently. The main benefit of temporal differences (red and black curves) is that high frequency temporal noise (distracting flicker) is reduced, in analogy to purely spatial gradient-domain rendering. In addition, we observe that mixed gradients (black) are more effective at high frequency noise suppression than using only temporal gradients (red). At equal computation costs, the temporal low frequency errors of both are higher than with only spatial gradients because of the additional overhead to compute the temporal and mixed gradients. This is not as perceptible as fast flickering, however, and the end result is more pleasing. In the figure, the MSE of conventional rendering without gradients is a constant $|\epsilon_F|^2 = 10 \times 10^{-3}$, which is long outside the range of the plots in the figure.

In Figure 5, we investigate what happens when the sampled spatial and temporal gradients have different variances. We slightly extend Equation 4 by allowing for separate variances for spatial gradients $|\epsilon_G|^2$, and temporal and mixed gradients $c|\epsilon_G|^2$, and denote their ratio by c. We plot curves for $c \in \{0.5, 1, 1.5, 2\}$, a typical range we observe in practice. The main observation is that even if temporal and mixed gradients have higher variance than the spatial ones, which may be due to fast object or camera motion, we still obtain some reduction in high frequency temporal error.

4 Extensions

4.1 Adaptive Sampling

Many previous authors have shown that distributing sampling efforts non-uniformly over the image plane to minimize the error of the output image may be beneficial. See Zwicker et al. [2015] for a recent survey. In addition, the energy of the gradients, and hence their variance, is usually sparsely distributed over the image [Olshausen and Field 1996]. This suggests that adaptive sampling may be even more effective in gradient-domain rendering, and that

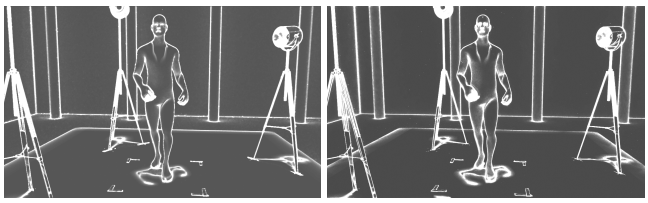


Figure 6: *Sampling distribution.* Left: *Average sampling map based on the variance of the spatial gradients over* 100 *runs at* 128 *samples per pixel in a frame of the* RUNNING MAN *scene.* Right: *Sampling map based on the variance of the L_1 reconstruction. We obtained the variance of the L_1 reconstructions by running the reconstruction* 100 *times on differently seeded input data at* 128 *samples per pixel each. The sampling maps look very similar, hence we use the estimated variance of the spatial gradients in practice.*

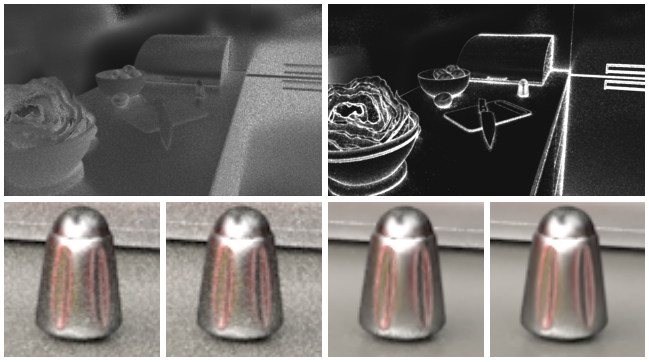
the sampling distributions for gradient-domain rendering will have stronger non-uniformity than ordinary adaptive sampling.

Ideally, to optimize output quality we should distribute samples according to the error of our reconstruction. As this is hard to estimate directly, we build on the following observation. We have found empirically that for reasonably high sampling rates, the variance of the L_1-reconstruction of individual frames correlates strongly with the sample variance of the spatial gradients. This is illustrated in Figure 6. While we cannot offer a theoretical justification for this, we further observe, and our results show, that distributing samples in each frame in proportion to the estimated variance of the spatial gradients yields significant improvements.

To get reliable variance estimates we distribute our samples in several batches: We distribute a first batch uniformly over the image, and use it to get an initial estimate of the sampling variance of the spatial gradients. We distribute all following batches such that they minimize the relative variance (variance divided by luminance of the pixel) of the gradients, and we also use them to update the variance estimates. Although this progressive adaptive sampling approach introduces bias, we consider it negligible in combination with our biased L_1 reconstruction. Usually the variance estimates of the gradients remain very noisy even for high sampling counts, in particular in scenes with specular paths. Hence we apply a 5×5 median filter on the variance estimates before using them to obtain sampling density maps. We further smooth the sampling density maps with a separable kernel $[0.1, 0.2, 0.4, 0.2, 0.1]$ and clamp the maximum values to eight times the average pixel sample count per sampling iteration to avoid sudden changes of sampling density.

Figure 7 compares the sampling distributions and final images achieved by the described algorithm to a standard path tracer with similar adaptive sampling based on the relative variance of its (regular) path samples. Adaptive sampling according to variance is clearly more effective in the gradient domain. This is due to the gradient sampler's ability to divert effort away from pixels where the underlying path space is smooth over the image coordinates. The regular sampler cannot exploit this, and is forced to a more uniform distribution by the variance along all path dimensions.

The primary sample space shift over time (Section 3.2) implies that the time shifted paths must be generated with the same random seeds as the paths in the base frame. In addition, the temporal offset frame must use the same sampling distribution as the base frame. To ensure this, we compute the sampling distribution map once for the base frame and store it on disk. To render the temporal offset frame, we read the sampling distribution map back from disk and distribute samples according to it.



PT, uniform PT, adaptive TGPT, uni. TGPT, adap.

Figure 7: Top: *Sampling distribution based on the relative primal image variance (left) compared to sampling distribution based on the relative gradient variance (right) on the* KITCHEN 1 *scene. We computed the densities with four sampling batches with an average of 256 samples per pixel each.* Bottom: *path tracing (PT) vs. our approach (TGPT) at equal render time, both with uniform and adaptive sampling using the respective sampling maps from the top row. Path tracing does not benefit from adaptive sampling here, while adaptive TGPT benefits from the sparsity in the sampling map and produces a visible improvement over uniform TGPT.*

4.2 Motion Vectors

Generating correlated pairs of path samples in two consecutive frames can be challenging. In presence of object or camera movement, the same primary sample space path shifted from one frame to the next, as described in Section 3.2, can hit very different geometry in these two frames. This leads to a high variance of the temporal differences. We mitigate this issue by taking into account scene and camera motion through per-pixel motion vectors that represent the motion of primary hit points from one frame to the next: instead of tracing the temporal offset path through the same pixel as the base path, we trace it through the corresponding pixel given by its motion vector. The intuition is that if we construct a temporal difference such that it follows the motion of the underlying object point, this often reduces the magnitudes of the temporal differences, and hence their variance. Of course, this procedure must be accounted for by the reconstruction algorithm. Note that this does not break consistency; good tracking decreases variance, but bad tracking does not break the algorithm. Many renderers, including Mitsuba that we build on, provide motion vectors for generating motion blur effects in post-processing.

4.2.1 Motion-aware Temporal Differences

In our pipeline, we obtain the temporal differences by rendering each frame twice, as a base and a temporal offset frame (Section 3.2). We render both with GPT, so each consists of a primary image and its spatial gradients. Then, temporal and mixed differences are obtained by simply subtracting the base from its succeeding temporal offset frame. Here we discuss how to include motion vectors in this process, as illustrated in Figure 8.

When rendering the base frame, we also compute motion vectors, and store them to disk. A motion vector represents the movement of a surface location between two frames, projected to the image plane and measured as a 2D pixel offset rounded to the closest integers. All motion vectors of a frame form a motion vector image. To render the temporal offset frame, we now read the motion vector image of the *previous* frame, compute the pixel correspondences

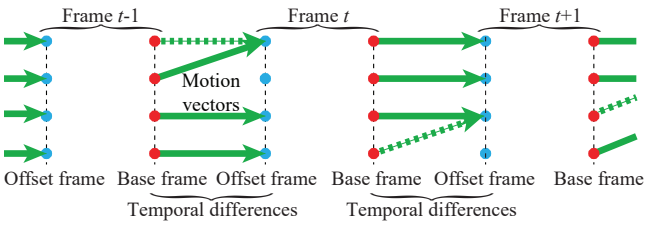


Figure 8: *Obtaining temporal differences with motion vectors. We render each frame twice using GPT, as a base (red) and an offset frame (blue circles as pixels). We obtain temporal differences as the difference between the offset frame at time $t+1$ and the base frame at time t, where pixel correspondences are given by the motion vectors (green arrows). Corresponding pixels use the same random seed to implement the primary sample space temporal shift (Section 3.2). Dotted green arrows indicate temporal differences that we ignore as described in Section 4.2.2.*

between the previous and the current frame, and then render the offset frame using the primary sample space shift (Section 3.2), i.e., using the primary sample space coordinates of the corresponding samples in the previous frame. If adaptive sampling is used, we also look up the sampling density in the previous frame.

After rendering, we average the base and offset frames to obtain the primary images $\mathbf{I}^g$ and the spatial gradients $\mathbf{I}^{dx}, \mathbf{I}^{dy}$. Next we warp each temporal offset frame onto the previous frame according to the motion vectors, that is, each pixel in a temporal offset frame is warped backward along its incident motion vector (green arrows in Figure 8). Temporal and mixed gradients $\mathbf{I}^{dt}$, $\mathbf{I}^{dxdt}$ and $\mathbf{I}^{dydt}$ are then obtained by subtracting the base frame (including its primary and spatial gradient images) from the warped, temporal offset frame. As shown in Figure 9, motion-aware temporal differencing generally significantly decreases the differences' magnitudes.

4.2.2 Reconstruction with Motion Vectors

Our conjugate gradient 3D Poisson solver computes $\mathbf{A}^T\mathbf{A}$ on the fly, which requires efficient access to the non-zero elements in rows of $\mathbf{A}$ and $\mathbf{A}^T$. Because we use motion vectors, however, the structure of $\mathbf{A}$ is modified, and not shift invariant any more. For simplicity, we explain how we take into account the motion vectors for the temporal constraints only. Computing the spatio-temporal constraints follows the same principles.

The temporal constraints consist of pairs of pixels (i,t) and $(\phi(i,t),t+1)$, where i is a pixel index, t a frame, and $\phi(i,t)$ is the index of the corresponding pixel in the frame $t+1$ given by the motion vector at pixel i and frame t. The row of $\mathbf{A}$ that represents the temporal gradient for pixel i at frame t has only two non-zero elements, given by the pixels i in frame t and $\phi(i,t)$ in frame $t+1$. Each row in the transpose $\mathbf{A}^T$ corresponds to a pixel i at frame t, and its non-zero elements correspond to all the constraints that (i,t) is involved in. This includes the temporal difference of some pixel j from frame $t-1$, which maps to i via the motion vectors, that is, $\phi(j,t-1)=i$. To find j we need to construct the inverse mapping ϕ^{-1}. For this mapping to be well-defined the motion map ϕ must be one-to-one, which unfortunately is usually not the case a priori.

We achieve invertibility of ϕ by also querying for a reverse motion map from frame $t+1$ to t. We require that whenever ϕ maps a pixel to the next frame, the reverse motion map maps the result back to the original pixel. We allow a tolerance of one pixel since the motion vectors are rounded to the nearest pixel. This tests that a pixel and its correspondence in the next frame represent the same object,

Figure 9: *Temporal differences with motion vectors.* Left: *Motion unaware temporal differences in a frame of the* BOOKSHELF *sequence at 512 samples per pixel.* Right: *Motion aware temporal differences for the same setup. The motion-aware temporal differences on the right are significantly smaller (gray represents zero).*

as pixels often represent objects that get occluded in the next frame. Including temporal difference constraints between different objects would lead to higher variance. We then check for any remaining cases of multiple pixels mapping onto one, and keep the ones that are mapped closest to the camera.

This means that not all pixels have a motion vector, hence some pixels do not have a temporal difference constraint. This is allowed since each pixel is still constrained by the primary image and the spatial gradients. With both the final ϕ and its inverse ϕ^{-1} accessible for the CUDA kernels, we quickly look up the non-zero elements in the rows of both $\mathbf{A}$ and $\mathbf{A}^T$ using $\phi(i,t)$ and $\phi^{-1}(i,t)$.

5 Implementation

We implemented our approach on top of the public GPT code by Kettunen et al. [2015] and made our code available on our web page. We built a spatio-temporal screened Poisson solver on the GPU, and process arbitrarily long animations by reconstructing overlapping sub-sequences. We use windows with 10 frames and overlap with the next window by 5 frames. This limits temporal data re-use to 10 frames, but we did not observe any benefits of larger temporal windows in practice. We produce the final output by smoothly blending the overlapping regions of the sub-sequences. Reconstructing 10 frames at a resolution of 1280×720 pixels takes around 15 seconds on an Nvidia GTX Titan and consumes roughly 2.6 GB of VRAM on the GPU. The only user parameter of the reconstruction is α (Equation 3), which we set to $\alpha=0.2$ for all our results.

6 Results and Discussion

In the following figures and the accompanying video we compare path tracing, gradient-domain path tracing, and our approach. All results are rendered with motion blur, which our approach handles without requiring any modifications, with an exposure time of half a frame. We further include comparisons of our approach with and without the proposed extensions (adaptive sampling and motion vectors).

Figure 10 visualizes the effect of temporal gradient sampling. At the top, we plot the luminance of a pixel over time for the different methods. At the bottom, we show a visual comparison using a space-time image of a short vertical 1D image segment over time. The plots and visualizations clearly show how our technique improves the temporal smoothness of the result. We also see the benefits of our extensions including adaptive sampling and motion vectors, which further reduce residual noise. The video and Figure 11 feature visual comparisons of our approach with conventional path tracing (PT), gradient-domain path tracing (GPT), and our approach (TGPT) at equal computation time. In the figure, we show one

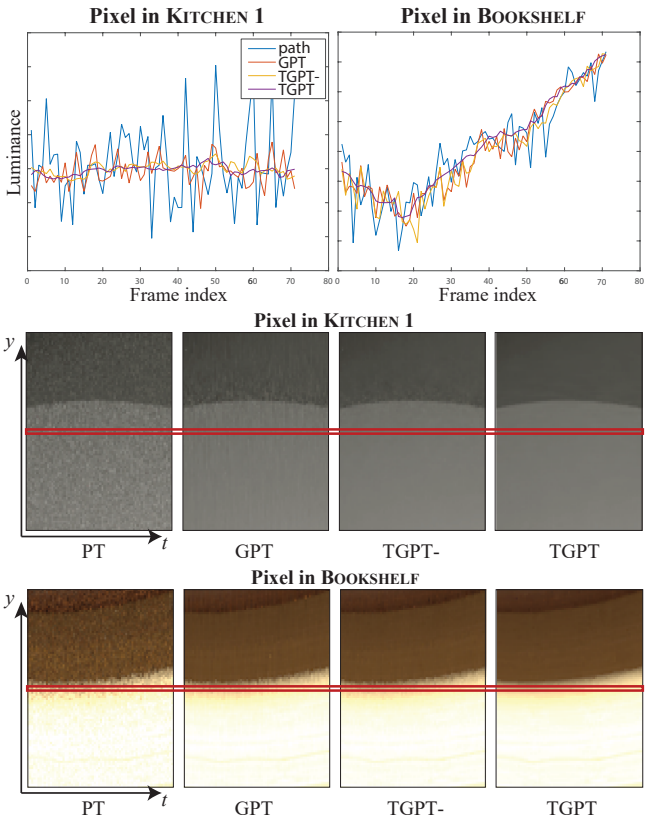


Figure 10: Top: *Temporal plots for one pixel in* KITCHEN 1 *and* BOOKSHELF. Bottom: *A* 95×1 *crop centered around the plotted pixels over time. We track the crops over time using the motion vectors of the central pixel, such that they are stationary relative to the surface hit point at the central pixel. Because rounding errors of the motion vectors accumulate over time, the image crops are still shifting a bit with respect to the central pixel. "TGPT-" denotes TGPT without any of the extensions discussed in Section 4.*

frame of the animation sequences for five test scenes (BOOKSHELF, SPONZA, RUNNING MAN, KITCHEN 1, KITCHEN 2). In addition to reducing temporal artifacts, TGPT clearly also suppresses spatial artifacts that are still visible in GPT.

To disentangle the effect of adaptive sampling and temporal differencing, the video includes a four-way comparison between spatial-only GPT, spatial-only GPT with adaptive sampling, TGPT without adaptive sampling, and TGPT with adaptive sampling using the RUNNING MAN scene. Comparing the two spatial gradient-domain algorithms, we observe that the clear improvement due to adaptive sampling at corners and other high-variance areas comes at the price of *increased* temporal flicker in the smooth areas. This is to be expected, as the sample budget is limited and adaptivity merely shuffles samples around. Temporal differencing reduces the flicker drastically, but without adaptivity, corners and glossy areas still suffer from variance. The combination of adaptation and temporal differencing produces clearly the best results.

We further study the effect of the speed of motion on TGPT in the video by comparing the RUNNING MAN animation at $1\times$, $2\times$ and $4\times$ the original animation speed. We can observe that the quality of the reconstruction degrades with increasing motion speed. The reason for this is three-fold: First, motion blur becomes more prominent, and regions where one single motion vector is insufficient to describe the motion of the underlying pixels become larger. Second, larger changes in scene geometry from frame to frame lead to less correlation between the base and offset paths, and hence to larger temporal differences. And third, larger regions with motion blur tend to make the adaptive sampling distribution more uniform, thus making adaptive sampling less effective.

Finally, the video includes an experiment where only the temporal differences are used in TGPT, that is, we omit spatial and spatio-temporal gradients. This method is attractive because it merely augments a simple path tracer with the temporal difference machinery, which operates on primary sample space and is very simple to implement. Without spatial differences, however, diffusion of the noise in the reconstruction can only occur along the time dimension. Hence, much longer temporal reconstruction windows are required to achieve significant noise reduction, which makes GPU memory management more challenging. The experiment in the video uses a temporal reconstruction window of 10 frames and we observe some noise reduction, but the noise appears *glued* to the surfaces and the result is visually unappealing.

6.1 Discussion and Limitations

Standard (spatial-only) gradient-domain path tracing only reduces noise in regions where the path throughput function is smooth under the shift mapping, i.e., where correlated spatially neighboring path pairs can be generated. Similarly, TGPT only reduces flickering in temporally smooth regions. This motivates our use of motion vectors to find similar pixel pairs in adjacent frames. Fast-moving or high-frequency moving geometry, however, increases variance as the temporal signal becomes less smooth. In addition, like all gradient-domain methods so far, TGPT cannot overcome the weaknesses of the underlying sampling method. For instance, it is poor at resolving caustics, since the underlying path tracer cannot sample them efficiently. Conversely, when the underlying sampler performs well, TGPT is very effective.

Several alternatives could be explored to reduce the variance of temporal differences, like adapting the half-vector preserving [Kettunen et al. 2015] or the manifold-walk based shift [Lehtinen et al. 2013] to the temporal domain. This would require information about the motion of each path vertex from one frame to the next, however, which is not provided in most existing renderers. We also experimented with using different weights for the spatial, temporal, and mixed gradients in the screened Poisson reconstruction, but did not obtain any significant improvements. We observed, however, that the local distribution of variances is rather different between the different types of constraints. This indicates that locally adapting the weights could be more effective than setting them globally.

7 Conclusions

We presented a temporal extension of gradient-domain rendering that significantly reduces temporal flickering artifacts compared to previous work. Our approach is unique in that it exploits temporal coherence already during Monte Carlo sampling, instead of imposing it solely in a post-process. We propose a simple temporal shift mapping to obtain temporal gradients using a primary sample space shift, which can be implemented with only small changes to an existing gradient-domain renderer. Our approach also uses mixed-spatio temporal gradients, and a frequency-analysis shows that they can further reduce temporal high frequency error. In addition, we proposed extensions to include adaptive sampling and motion vectors, which effectively boost the quality of our results. In the future we would like to investigate more sophisticated spatial and temporal shift mappings to further improve the quality of gradient-domain

rendering. We will also investigate other gradient stencils that may provide additional benefits over our current spatial, temporal, and mixed stencils. Finally, it would be interesting to analyze the benefits and limits of adaptive sampling from a theoretical perspective.

Acknowledgments

We thank Sampo Rask for RUNNING MAN, and Benedikt Bitterli and blendswap.com user Jay-Artist for KITCHEN 2. This work was supported by the Swiss National Science Foundation, projects 143886 and 163045, NSF project 1451830, and the Academy of Finland, grant 277833.

References

BONNEEL, N., SUNKAVALLI, K., TOMPKIN, J., SUN, D., PARIS, S., AND PFISTER, H. 2014. Interactive intrinsic video editing. *ACM Trans. Graph. 33*, 6 (Nov.), 197:1–197:10.

BONNEEL, N., TOMPKIN, J., SUNKAVALLI, K., SUN, D., PARIS, S., AND PFISTER, H. 2015. Blind video temporal consistency. *ACM Trans. Graph. 34*, 6 (Oct.), 196:1–196:9.

CHEN, S. E. 1990. Incremental radiosity: An extension of progressive radiosity to an interactive image synthesis system. *SIGGRAPH Comput. Graph. 24*, 4 (Sept.), 135–144.

GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., AND BATTAILE, B. 1984. Modeling the interaction of light between diffuse surfaces. *SIGGRAPH Comput. Graph. 18*, 3 (Jan.), 213–222.

HAVRAN, V., DAMEZ, C., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2003. An efficient spatio-temporal architecture for animation rendering. In *Proceedings of the 14th Eurographics Workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EGRW '03, 106–117.

KELEMEN, C., SZIRMAY-KALOS, L., ANTAL, G., AND CSONKA, F. 2002. A simple and robust mutation strategy for the metropolis light transport algorithm. *Computer Graphics Forum 21*, 3, 531–540.

KELLER, A., FASCIONE, L., FAJARDO, M., GEORGIEV, I., CHRISTENSEN, P., HANIKA, J., EISENACHER, C., AND NICHOLS, G. 2015. The path tracing revolution in the movie industry. In *ACM SIGGRAPH 2015 Courses*, ACM, New York, NY, USA, SIGGRAPH '15, 24:1–24:7.

KETTUNEN, M., MANZI, M., AITTALA, M., LEHTINEN, J., DURAND, F., AND ZWICKER, M. 2015. Gradient-domain path tracing. *ACM Trans. Graph. 34*, 4 (July), 123:1–123:13.

LEHTINEN, J., KARRAS, T., LAINE, S., AITTALA, M., DURAND, F., AND AILA, T. 2013. Gradient-domain metropolis light transport. *ACM Trans. Graph. 32*, 4 (July), 95:1–95:12.

LI, T.-M., WU, Y.-T., AND CHUANG, Y.-Y. 2012. Sure-based optimization for adaptive sampling and reconstruction. *ACM Trans. Graph. 31*, 6 (Nov.), 194:1–194:9.

MANZI, M., ROUSSELLE, F., KETTUNEN, M., LEHTINEN, J., AND ZWICKER, M. 2014. Improved sampling for gradient-domain metropolis light transport. *ACM Trans. Graph. 33*, 6 (Nov.), 178:1–178:12.

MANZI, M., KETTUNEN, M., AITTALA, M., LEHTINEN, J., DURAND, F., AND ZWICKER, M. 2015. Gradient-domain bidirectional path tracing. In *Proc. Eurographics Symposium on Rendering*.

MCCOOL, M. D. 1999. Anisotropic diffusion for monte carlo noise reduction. *ACM Trans. Graph. 18*, 2 (Apr.), 171–194.

MEYER, M., AND ANDERSON, J. 2006. Statistical acceleration for animated global illumination. *ACM Trans. Graph. 25*, 3 (July), 1075–1080.

MOON, B., CARR, N., AND YOON, S.-E. 2014. Adaptive rendering based on weighted local regression. *ACM Trans. Graph. 33*, 5 (Sept.), 170:1–170:14.

NIMEROFF, J., DORSEY, J., AND RUSHMEIER, H. 1996. Implementation and analysis of an image-based global illumination framework for animated environments. *IEEE Transactions on Visualization and Computer Graphics 2*, 4 (Dec.), 283–298.

OLSHAUSEN, B., AND FIELD, D. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature 381*, 607–609.

ROUSSELLE, F., MANZI, M., AND ZWICKER, M. 2013. Robust denoising using feature and color information. *Computer Graphics Forum 32*, 7, 121–130.

RUSHMEIER, H. E., AND WARD, G. J. 1994. Energy preserving non-linear filters. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '94, 131–138.

SCHERZER, D., YANG, L., MATTAUSCH, O., NEHAB, D., SANDER, P. V., WIMMER, M., AND EISEMANN, E. 2012. Temporal coherence methods in real-time rendering. *Computer Graphics Forum 31*, 8, 2378–2408.

SEN, P., AND DARABI, S. 2012. On filtering the noise from the random parameters in monte carlo rendering. *ACM Trans. Graph. 31*, 3 (June), 18:1–18:15.

SMYK, M., KINUWAKI, S.-I., DURIKOVIC, R., AND MYSZKOWSKI, K. 2005. Temporally coherent irradiance caching for high quality animation rendering. *Computer Graphics Forum 24*, 3, 401–412.

TAWARA, T., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2004. Exploiting temporal coherence in final gathering for dynamic scenes. In *Proc. Computer Graphics International, 2004*, 110–119.

VEACH, E., AND GUIBAS, L. J. 1997. Metropolis light transport. In *Proc. of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '97, 65–76.

WARD, G. J., RUBINSTEIN, F. M., AND CLEAR, R. D. 1988. A ray tracing solution for diffuse interreflection. *SIGGRAPH Comput. Graph. 22*, 4 (June), 85–92.

ZIMMER, H., ROUSSELLE, F., JAKOB, W., WANG, O., ADLER, D., JAROSZ, W., SORKINE-HORNUNG, O., AND SORKINE-HORNUNG, A. 2015. Path-space motion estimation and decomposition for robust animation filtering. *Computer Graphics Forum 34*, 4, 131–142.

ZWICKER, M., JAROSZ, W., LEHTINEN, J., MOON, B., RAMAMOORTHI, R., ROUSSELLE, F., SEN, P., SOLER, C., AND YOON, S.-E. 2015. Recent advances in adaptive sampling and reconstruction for monte carlo rendering. *Computer Graphics Forum 34*, 2, 667–681.

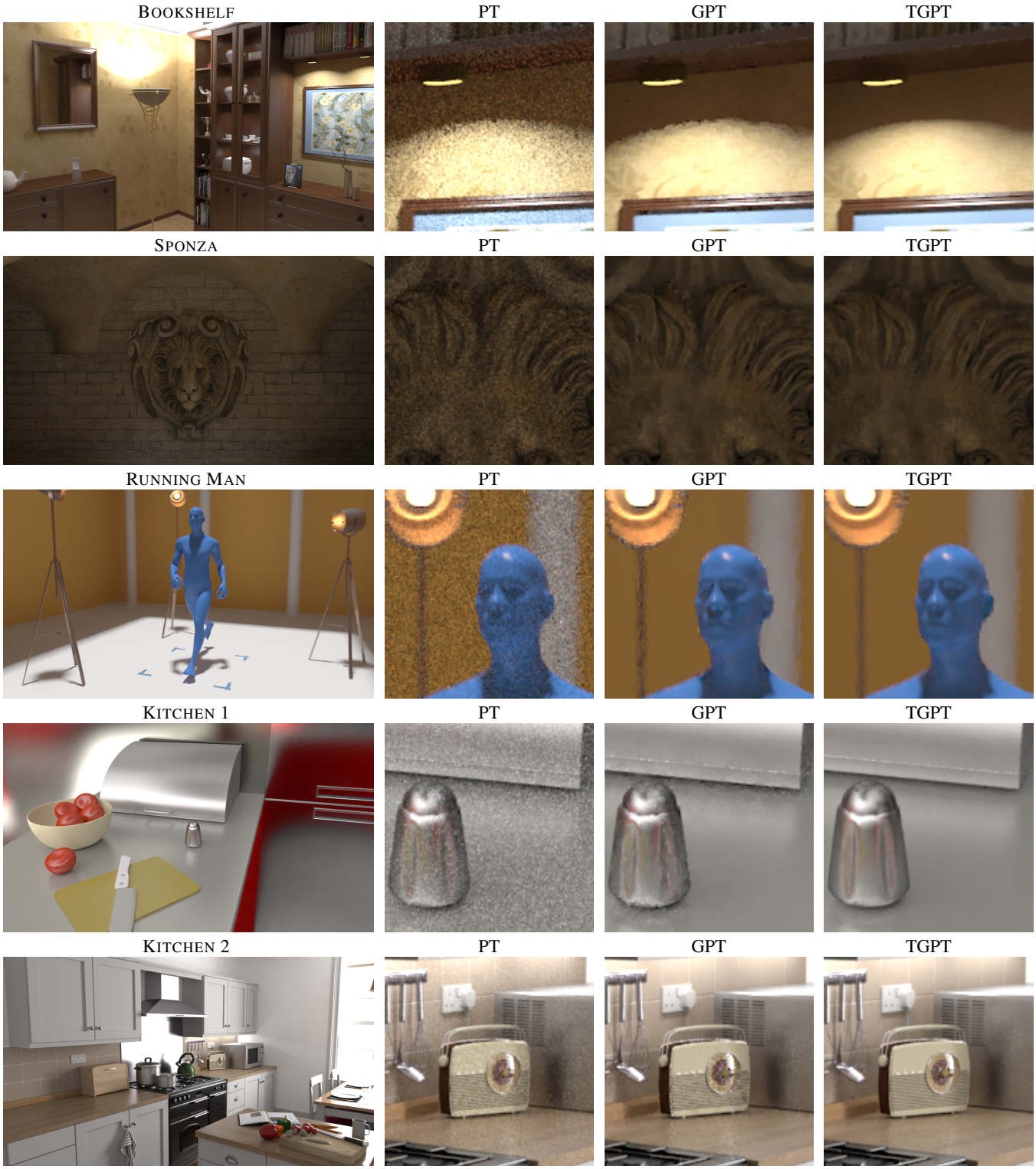


Figure 11: *Comparison of our approach with conventional path tracing (PT), gradient-domain path tracing (GPT), and our proposed temporal gradient-domain algorithm (TGPT) at equal computation time. We show one frame of the animation sequences for our five test scenes, and a close-up each. In addition to reducing temporal artifacts, TGPT clearly also suppresses residual spatial artifacts that are still visible in GPT. We rendered the first three scenes, from top to bottom, with 32 samples per pixel (spp) using GPT and TGPT, and 80 spp using PT, which amounts to equal time given the 2.5× overhead of GPT over PT. The last two scenes have 256 spp for GPT and TGPT, and 640 spp for PT. TGPT splits the samples per frame into two halves to render each frame twice, as temporal base and offset frame (Figure 8).*